Vis Comput (2016) 32:371–381
DOI 10.1007/s00371-015-1185-9

ORIGINAL ARTICLE

Animated visualization of spatial–temporal trajectory data for air-traffic analysis

Stefan Buschmann[1] · **Matthias Trapp**[1] · **Jürgen Döllner**[1]

Published online: 28 November 2015

Abstract With increasing numbers of flights worldwide and a continuing rise in airport traffic, air-traffic management is faced with a number of challenges. These include monitoring, reporting, planning, and problem analysis of past and current air traffic, e.g., to identify hotspots, minimize delays, or to optimize sector assignments to air-traffic controllers. To cope with these challenges, cyber worlds can be used for interactive visual analysis and analytical reasoning based on aircraft trajectory data. However, with growing data size and complexity, visualization requires high computational efficiency to process that data within real-time constraints. This paper presents a technique for real-time animated visualization of massive trajectory data. It enables (1) interactive spatio-temporal filtering, (2) generic mapping of trajectory attributes to geometric representations and appearance, and (3) real-time rendering within 3D virtual environments such as virtual 3D airport or 3D city models. Different visualization metaphors can be efficiently built upon this technique such as temporal focus+context, density maps, or overview+detail methods. As a general-purpose visualization technique, it can be applied to general 3D and 3+1D trajectory data, e.g., traffic movement data, georeferenced networks, or spatio-temporal data, and it supports related visual analytics and data mining tasks within cyber worlds.

Keywords Spatio-temporal visualization · Trajectory visualization · 3D visualization · Visual analytics · Real-time rendering

✉ Stefan Buschmann
stefan.buschmann@hpi.de

[1] Hasso Plattner Institute, University of Potsdam, Prof.-Dr.-Helmert-Str. 2-3, 14482 Potsdam, Germany

1 Introduction

In spatio-temporal data analysis and visualization [5], research focuses on the development of methods and techniques that facilitate the human perception and understanding of temporal aspects of spatial data, e.g., concurrency and temporal order of events [6]. In addition to static depictions, animation and interactive exploration provide effective means for presenting, analyzing, and understanding spatial and temporal aspects.

In recent years, visual analytics has emerged as a broad research field "relating to the theory, design, and development of interactive visual interfaces for the purposes of data exploration, data analysis, sense making, and decision making" [10]. Visual analytics methods are especially useful when dealing with large-scale, complex data sets: by allowing users to explore data sets and to interact with the applied visualization and analysis methods, the human cognition and its capability to recognize patterns is effectively employed.

1.1 Visual analytics for 3D trajectory data

Various application and research domains deal with large complex spatio-temporal data sets, which can often be interpreted as connected, attributed lines or graphs. This includes infrastructural data such as telecommunication or power grids, bio-information such as molecules or DNA strings [23], spatio-temporal events, and movement data [17]. Visualization of such data is a demanding task because it includes rendering of complex data (a large number of data attributes), structural information (connection between data items), as well as the depiction of temporal aspects (i.e., changes over time).

Within this field, analysis and visualization of movement data represent one fundamental category. In particular, the

visualization of movement trajectories, often embedded in a geographic context, represents a key functionality. This includes the representation of spatial and temporal aspects of movements as well as depicting additional data attributes of complex attributed trajectories. General challenges include: (1) to handle the data complexity, (2) to detect and communicate spatial and temporal phenomena, and (3) to handle the typically large data sets while maintaining flexibility and interactivity for, e.g., the analyst user.

1.2 Contributions

In this paper, we propose a real-time animated visualization technique for massive complex movement trajectories embedded in a 3D virtual environment (Fig. 1). It supports 3D trajectory visualization, including a configurable mapping to visualize static attributes and dynamics, and aggregation techniques such as the generation of 2D density maps. To achieve visualization and exploration at interactive frame rates, all stages of the visualization pipeline [39] are implemented based on GPU techniques, including filtering, mapping, and rendering. This enables interactive spatio-temporal filtering, configurable mappings of attributes to visual properties, and real-time rendering of large data sets. Several visualization metaphors such as temporal focus+context [9], density-based analysis [33], and space–time cube visualization [20] are implemented and evaluated on the use case of air-traffic data. To summarize, this paper makes the following contributions:

1. Description of a fully GPU-based visualization pipeline, including filtering, dynamic mapping, and real-time rendering of complex trajectory data.

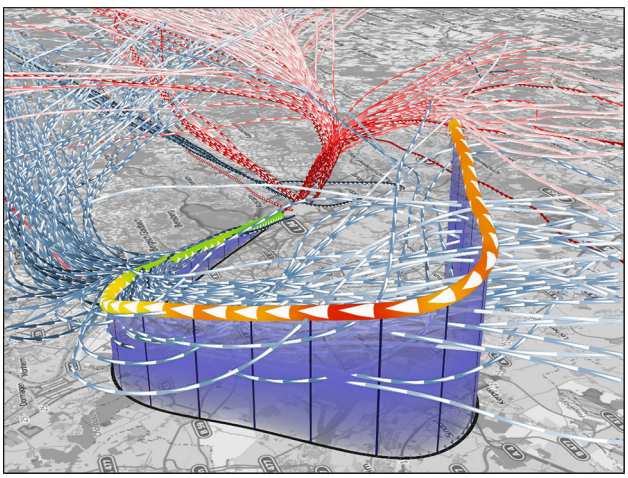


Fig. 1 Exemplary visualization of aircraft trajectories using color mapping, texturing, and animation. The image shows departing (*red*) and arriving (*blue*) aircrafts. For a selected trajectory, *animated arrows* depict the direction, while acceleration is visualized by *color*

2. Application of the described visualization pipeline to implement different visualization techniques for the interactive visualization of movement data.

The remainder of this paper is structured as follows. Section 2 describes the application context and data set used in this work and states related challenges and requirements. Section 3 reviews related work. Section 4 presents the concept of our GPU-based visualization pipeline. Section 5 demonstrates application examples. Section 6 contains performance analysis and discussion. Finally, Sect. 7 outlines conclusions and future research directions.

2 Problem statement

This paper presents an approach for interactive visualization and analysis of 3D air-traffic trajectories. In this context, processing and visualization of large sets of 3D trajectories at interactive frame rates, as well as providing tools for interactive analysis and exploration to analysts, are the main challenges. In the following section, the application context, use cases, the resulting challenges, and technical requirements are described.

2.1 Application context

The field of air-traffic management (ATM) comprises operational tasks, i.e., air-traffic control (ATC), as well as a broader spectrum of management tasks, in particular, airspace, flow, and capacity management. While ATM is mainly concerned with monitoring and managing the ongoing situation in air traffic, it also includes the analysis of past situations and the planning for the future, i.e., by modifying flight routes or adjusting regulations. In this context, the analysis of air plane movements can be an important tool for analyzing and understanding past events, for assessing current events, and finally for drawing conclusions for the future. An objective may be to analyze past events and derive insights from it, as to understand the cause of events and learn how to prevent them or optimize processes. Exemplary applications, questions, and interests for the analysis of air plane movements include:

(A.1) Flight-path analysis By investigating the distribution of movements over time and space, areas with higher and lower traffic volumes can be identified. From this, common flight routes, which are used often, can be derived. Trajectories can be categorized according to those flight routes.

(A.2) Classification A comparison of flight routes allows for identifying similarities and differences between different flights. By correlating them to other flight attributes (e.g., departing and arriving flights or different aircraft types), a classification of flights can be achieved.

(A.3) Pattern discovery A further goal is the identification of typical and untypical movement behaviors (i.e., detect outliers) and if possible correlate them to spatial or temporal circumstances. In this context, clustering methods [2,3] can be applied to categorize trajectories.
(A.4) Adherence to safety regulations Using additional data such as flight routes, air-traffic sectors, or safety regulations, flight traffic that violates those can be identified. A goal is to find reasons or circumstances of such violations and if possible to compare consequences of administrative or regulatory actions regarding air safety violations.
(A.5) Comparison of routing scenarios For air-traffic planning, one requirement is to compare alternative scenarios. For example, the results of changing flight routes, or the consequences of how a new air port is designed, can be examined by simulating and comparing the resulting movement trajectories.

2.2 Data set and data acquisition

The data set used in this paper comprises true 3D trajectories of aircraft movements and is a typical example of aircraft movement data. It originates from RADAR tracking in the vicinity of an airport and includes departing and arriving IFR (instrument flight rules) flights. The tracking data have a temporal resolution of four seconds and a range of about 50–70 km. A single trajectory has a duration of 5–10 min with approximately 75–150 sample points. For a duration of one month, the data set consists of a total number of 12,500 trajectories and more than 1,500,000 sample points.

The data set consists of two types of data items: nodes and trajectories. A *node* denotes an individual sample point, while a *trajectory* denotes the movement of an aircraft by a consecutive list of nodes. In addition to spatial and temporal information, the flight data contain attributes *per-trajectory* and *per-node*. Per-trajectory attributes include a flight identification (ICAO call-sign), the type of aircraft, and metadata about the flight from the airport (i.e., arrival or departure, and runway designations). Per-node attributes include a time stamp, the current geographical location, as well as the current height and velocity over ground.

2.3 Visualization of 3D trajectories

To support the analysis of the described data set with respect to the tasks and questions mentioned before, interactive visualization and visual analytics tools can be applied. This requires tools that support real-time visualization of large numbers of 3D trajectories, their spatial and temporal features, as well as their attributes. In addition, interactive exploration in both, the spatial and the temporal domain, needs to be supported. This enables users to explore a data set, identify spatial and temporal phenomena, correlate them to trajectory attributes and thereby gain insight into the data.

Therefore, visualization methods have to be designed to work on large amounts of data, to find patterns and phenomena in past events. For an international airport, this often exceeds 10,000 individual trajectories per month. Extending the scope of analysis either spatially or temporally, e.g., by applying it to real-time data, results in a massive amount of data that have to be processed and visualized. Another challenge is the demand for visual analytic systems, in which users can explore data interactively. Finally, the inherent 3D nature of the data set poses additional challenges to a visualization system.

2.4 Challenges in 3D air-traffic visualization

In the discipline of spatio-temporal analytics, in particular, the analysis of moving objects, traditional visualization is often based on 2D movement representations. In our use case, though, movements need to be visualized in 3D space, since their movement characteristics significantly depend on all three dimensions. For example, focusing on aircraft trajectories, the detection of flight routes, common flight corridors, as well as spatial and temporal hotspots demand for a true 3D analysis. Therefore, the actual 3D positions of aircrafts have to be maintained and expressed in a visualization. However, temporal aspects have to be conveyed as well. This poses a number of major challenges for 3D visualization techniques dealing with massive spatio-temporal data:
(C.1) Visualization of temporal aspects Since three axes are used to display the spatial components of the data itself, no axis is left to express the temporal information. Therefore, temporal information has to be mapped to other visual properties such as color, width, or texture, or has to be visualized using other forms, e.g., by means of animation.
(C.2) Perspective foreshortening When using perspective projections of 3D scenes, spatial positions appear distorted in the image, making perception of height and size prone to misinterpretation. The effect is inevitable, in particular when using non stereoscopic displays, but must be accounted for, since it can disturb the cognition of the displayed data.
(C.3) Occlusion and clutter Occlusion [11] poses another major problem for 3D visualization of massive movement data (Fig. 2a). It is caused by overlapping trajectories, occluding the objects behind them. A large number of objects displayed at the same time also leads to visual clutter (Fig. 2b).

2.5 Requirements for visualization systems

Because of the aforementioned challenges, visualization methods for this kind of data tend to be (1) specialized and strongly tied to a specific application domain, and (2) often not flexible in terms of adaptation and re-configuration of

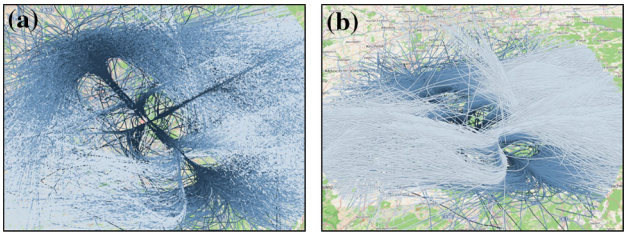


Fig. 2 Comparison between 2D and 3D visualization of a massive set of movement trajectories. **a** 2D movement trajectories. **b** 3D movement trajectories

visualization parameters. However, interactive systems for visual analytics require a high degree of flexibility of visualization methods, and high performance with respect to data processing and rendering. To enable users to interactively explore and analyze data sets, both, data-filtering and attribute mapping [39], have to be configurable in real time which results in the following requirements [38]:
(R.1) Flexibility For effective analysis, a visualization pipeline must be highly configurable to visualize, combine, and correlate different aspects of the data.
(R.2) Memory consumption To enable the visualization of large data sets, memory consumption for (mapped) data representation should be minimal.
(R.3) Update performance The modification of mapping or visualization parameters can require the recreation of geometry, which has to be transferred to the GPU. To maintain interactive frame rates, update costs must also be small.
(R.4) Rendering performance Rendering time must be low enough to support the visualization of large data sets. In addition, methods such as batching and level-of-detail [28] should be applied.
These requirements affect the choice and adaptation of individual visualization, analysis, and rendering techniques, which can be applied for interactive visualization and analysis systems for massive movement data.

3 Related work

Existing visualization approaches for spatio-temporal data are often either inherently two dimensional or use the third dimension to display information other than the spatial 3D component of the input data (e.g., time or other data attributes). This section reviews major contributions w.r.t. the presented application context.

Air-traffic data The visualization of movement trajectories is important in the context of air-traffic data. Hurter et al. have developed an interactive tool for the exploration of massive amounts of aircraft trajectories [17] using brushing and connected viewports. Further, image-based approaches have been used to visualize and analyze aircraft trajectories: Hurter et al. have applied a temporal bundling approach to visualize trajectories and dynamic graphs [15], and extended the approach to develop visual analytics methods for the analysis of large data sets of air-traffic trajectories [14,19]. As a specific analysis task, they proposed a method to extract wind parameters from aircraft trajectories [13]. In this work, some of these techniques are combined and extended with the main focus on interactive 3D visualization of massive air-traffic trajectories.

Trajectory and attribute visualization To visualize attributes of complex data, the concept of visual glyphs [30] can be used. Glyph-based visualizations have been applied to convey time-dependent data, e.g., time series, in the context of geographical maps [34,36]. Tominski et al. have used a wall-like approach to visualize spatio-temporal data in a 2D geographical context [35]. Nienhaus et al. used dynamic glyphs to depict dynamics following visual art and graphic design principles [31]. In the context of movement trajectories, the visualization of attributes is an important functionality. To visualize attribute values together with spatial and temporal information, Tominski et al. used a stacking-based approach [37]. Andrienko et al. have applied a similar approach to the space–time cube to visualize attributes of trajectories following similar routes [4]. The visualization techniques presented in this paper use a similar approach for attribute visualization, using a real-time mapping approach for interactive contexts.

Space–time cube The space–time cube [12] is based on a three-dimensional representation of space and time: X- and Y-axes are used for spatial components, while time is mapped to the Z axis. Therefore, it is especially suitable for visualizing two-dimensional data with a temporal component, but may not be easily applicable to visualize true 3D data. In recent years, the space–time cube has become a popular visualization method in the field of geovisualization and visual analytics [20], and several frameworks based on the space–time cube have been developed [7,21]. Kristensson et al. have conducted a user study on the space–time cube [22], identifying tradeoffs regarding the use of a space–time cube for presenting spatio-temporal data to users. Their results indicate that the space–time cube can support the comprehension of complex spatio-temporal data. Kveladze et al. have developed a methodological framework for evaluating the usability of the space–time cube [24]. The space–time cube concept is applied in our work as one available mapping configuration, to support the detailed temporal analysis of movement trajectories.

Density maps Density maps can be applied to aggregate movements using a regular grid. Willems et al. used kernel density estimation to aggregate vessel movements in a harbor [40], applying different kernel sizes to distinguish between

historical and current movements. This enables users to identify current movements and detect untypical and potentially dangerous movement behavior. Scheepens et al. introduced interactive density maps by utilizing GPUs, encoding arbitrary attributes by modifying the kernel attributes and introduced an interactive approach for selecting subsets of trajectories [33]. Since density maps are a valuable component for pattern recognition in movement data, we integrated them into our real-time visualization pipeline.

4 Architecture of visualization pipeline

Visual analytic systems for movement data need to be highly interactive, enabling users to configure filtering, mapping, and visualization parameters in real time to explore and analyze massive spatio-temporal data sets.

4.1 Conceptual overview

Therefore, we propose a visualization pipeline, in which all stages (filtering, mapping, and rendering) are implemented on the GPU (Fig. 3). In particular, complex input data, consisting of attributed nodes and trajectories, is uploaded directly to the GPU, including all attributes available for filtering and mapping. Instead of creating complex geometry on the CPU and uploading it onto the GPU, mapping and geometry creation are deferred and executed on the GPU during rendering. This reduces the memory consumption of the data and enables interactive configuration of filtering, mapping, and geometry creation at runtime.

In a *preprocessing* stage, input data are transformed into a representation suitable for rendering and subsequently uploaded to the GPU. This comprises a selection of input attributes that are required for mapping, as well as the computation of derived attributes such as results of analysis functions, classification, or aggregation. In addition to that, a set of *mapping configurations* are defined and uploaded to the GPU. They define the entire visualization settings, including the type of geometry, tessellation factor (number of triangles per segment), the mapping of input values to visual properties, as well as additional appearance options (e.g., textures, color maps, or geometric constraints).

During rendering, the mapping configuration is determined for each vertex individually and applied to visualize the respective data item (Fig. 4). The concept can be implemented fully hardware-accelerated using vertex, geometry, and fragment shaders [18]:

1. A vertex shader is responsible for selecting an appropriate mapping configuration for each input vertex. Based on the attribute data, current user selection, and a level-of-detail parameter, it selects a configuration, applies the mapping from data attributes to visual properties, and finally passes the configuration on to the geometry shader. Filtering is applied by omitting the generation of geometry for items that are excluded by the current filter options.
2. A geometry shader generates the appropriate geometry for each vertex according to the selected mapping. This stage is computationally expensive, yet it constitutes the flexibility of the approach (R.1). For example, it is possible to create completely different types of geometries such as lines, tubes, or spheres, based on attributes of the input data (Fig. 6), and this mapping can even be modified at runtime or by user interaction at a per-frame level.
3. During rendering, the generated geometry is rasterized. In addition, stylization and post-processing effects are applied to improve the quality of the generated visualization artifact.

To summarize, this approach enables performing complex filtering and mapping operations implemented entirely on the GPU (R.1). Further, it reduces data transfer between the application stage and the GPU, thus increasing update performance (R.3) and yielding reduced memory consumptions (R.2).

4.2 Interactive filtering modes

Filtering of input data helps to focus a user's attention on the selected items, reduces visual clutter (C.3), and improves performance by reducing the amount of data that needs to be

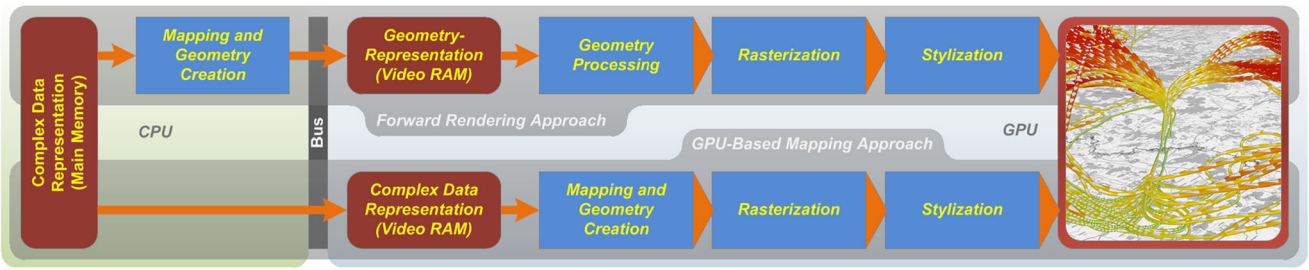


Fig. 3 Comparison of two rendering pipelines for the visualization of complex data: a traditional CPU-based approach for geometry creation (*top*), and the presented GPU-based approach (*bottom*)

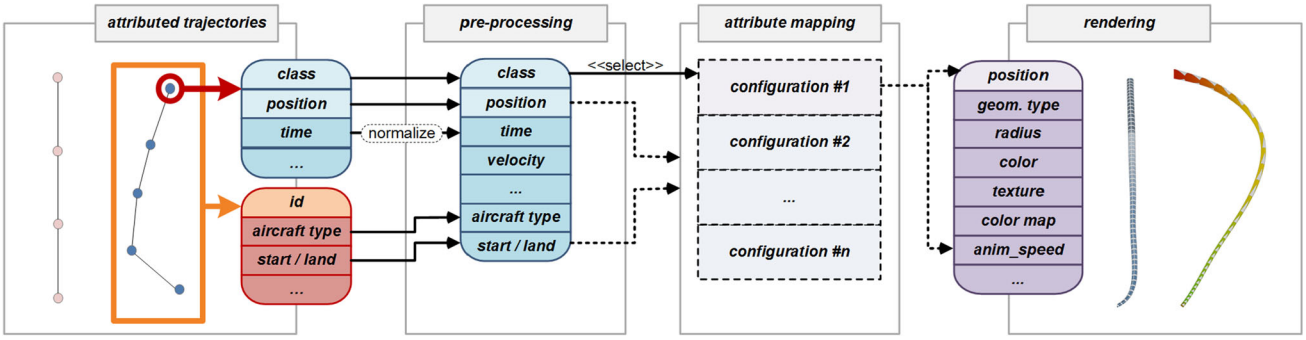


Fig. 4 Schematic mapping overview: attributed nodes and trajectories constitute the input data items for the mapping stage. The primary attribute *class* selects the actual mapping configuration that is applied. During attribute mapping, input attributes are mapped to visual properties of the visualization technique

processed and visualized. For this, a user selects the subset of data that is subsequently visualized, while the remaining items are either omitted or visualized less prominently. In the presented pipeline, three filtering concepts are supported: *spatial*, *temporal*, and *attribute-based* filtering. All three options can be combined and controlled interactively by the user.

Spatial filtering Data items are filtered by their geographic position, to restrict the analysis to a specific region-of-interest and thereby reduce the amount of data that needs to be processed and visualized.

Temporal filtering This method filters items by their temporal information, accepting items that are located within a certain time span or correspond to repeating time intervals, such as specific day times. Such a time span is defined by user interaction (e.g., using a virtual time line). Further, multiple time spans can be selected and used by the visualization methods, e.g., for a visual comparison of differences in density maps, or to highlight different temporal foci in a focus+context visualization [9].

Attribute-based filtering Items are filtered based on attribute values, e.g., using minimum and maximum values to define a range in that items are visible. This is a useful analysis tool to focus on items that related to specific criteria and to reduce the data set by omitting all other items. Attribute filters can be applied to all data items, i.e., nodes as well as trajectories. Since filtering takes place on the GPU, users can redefine all filters interactively and the result is visualized immediately. Based on the filtering results, the trajectory data are converted to rendering primitives in a subsequent mapping stage that is entirely performed on the GPU.

4.3 Hardware-accelerated attribute mapping

To visualize attributed trajectories, data items have to be transformed into geometric primitives, and a mapping of data attributes to visual properties needs to be performed. This is performed by selecting a mapping configuration for each data item, which describes the parameterization of geometry and mapping, and applying it to its attributes. The choice of the mapping configuration is not static, but depends on several input sources, which are explained in the following.

Input data and classification Most importantly, the mapping configuration can depend on the data attributes. These can be used to classify data items and apply different mappings for each distinct class. For example, individual configurations can be selected to differentiate between departing and arriving aircrafts, or between aircraft types. Further, classifications can be used to emphasize on specific sample points within a single trajectory, e.g., to identify turns or phases of acceleration.

Classifications can also be applied to combine different styles in a single visualization. For example, Fig. 5 shows visualization variants for displaying acceleration values. In Fig. 5a, acceleration is visualized directly by color encoding. In Fig. 5b, the same data attribute is used to categorize into phases of high and low acceleration. These are rendered using two different mapping configurations: high acceleration is visualized by tubes, while the remainder of a trajectory is represented by spheres. In addition, speed values are encoded in color and direction is mapped to a texture. Therefore, this classification adds additional information to the visualization by enabling the differentiation between phases of high and low acceleration, while consistently mapping speed to color.

Level-of-detail mapping After a mapping configuration has been selected based on the item's data, the mapping system applies level-of-detail by selecting different configurations based on the current distance of the virtual camera to the visualized item. This provides the option to exchange or modify the mapping configuration for each level-of-detail, e.g., to reduce visual details or decrease the tessellation factor for objects that have a high distance to the virtual camera.

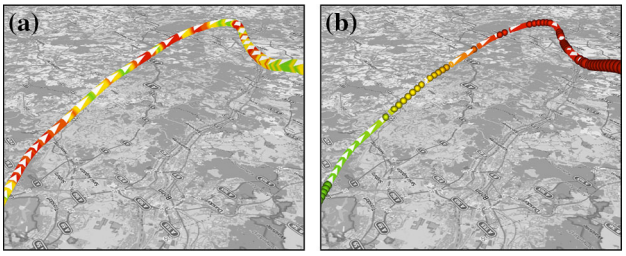


Fig. 5 Classification examples. **a** Visualization of acceleration, encoded in *color*. **b** Visualize high (tubes) and low (spheres) acceleration

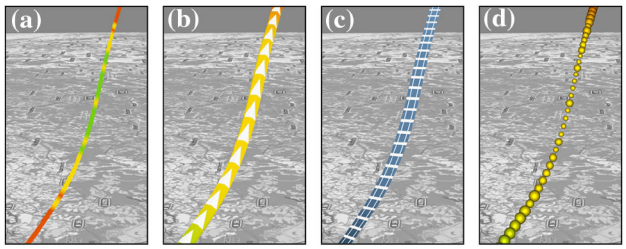


Fig. 6 Supported basic geometry types. **a** Lines. **b** Tubes. **c** Ribbons. **d** Spheres

Filtering parameter The results of the filtering stage can also be used to influence mapping configurations. For example, for focus+context visualizations, different configurations can be used for selected (*focus*) and unselected items (*context*), or to distinguish between multiple defined foci.

User selection A mapping configuration can also be influenced by user interaction, for example, by modifying mapping configurations to visually highlight those items that are currently selected by the user (C.3).

4.4 Image synthesis

Subsequent to the mapping stage, geometric primitives are created on the GPU and configured according to the chosen mapping parameters. In the next stage, this geometry is rendered using rasterization. This section covers rendering techniques and concepts incorporated in the presented visualization pipeline to serve real-time rendering constraints and visual quality of the visualization artifacts.

Trajectory and glyph rendering The central functionality of the presented visualization approach consists of the efficient rendering (R.4) of 3D movement trajectories and their attributes. For this, several geometric primitives (lines, ribbons, tubes, or spheres) can be used. To visually communicate the values of attributes, they are mapped to visual properties such as color, line width, or the diameter of tubes or spheres. These geometric representations are supplemented by texture mapping [1] and animation, which can for example be applied to convey the direction and speed of aircrafts using animated arrow textures (C.1).

Figure 6 shows the basic geometry types provided by the visualization technique. An often used geometry type are lines (Fig. 6a), providing color as a visual property. Further, tubes (Fig. 6b) support radius, color, a texture map, and texture animation. The same properties are supported by ribbons (Fig. 6c), with the exception of having a width parameter instead of a radius. Furthermore, spheres (Fig. 6d) can be used to visualize the attributes of individual sample points, providing color and radius as visual properties.

To increase performance, the geometric primitives are rendered mainly with billboard techniques [1] using local shading models to create the impression of the specific cylindrical or spherical shapes, while reducing the amount of geometry and tessellation that needs to be processed on the graphics board.

Generation of visual cues Due to the nature of 3D rendering and perspective projections, position and height can be hard to perceive (C.2). To improve this, visual cues can be applied. For example, shadows can facilitate the perception of trajectory positions with respect to the underlying map. In addition, drop-lines or fences can improve the perception of height along the course of a trajectory. These visual cues can be created automatically during geometry generation, and can be controlled by the dynamic mapping configuration, e.g., to enable visual cues only for detailed inspection of the selected trajectory, while omitting them for the other trajectories, as shown in Fig. 1.

Generation of density maps To support the generation and visualization of aggregated data representations, density maps can be computed in real time, using the filtering, mapping, and rendering pipeline as described before. The synthesis of density maps is performed using off-screen rendering in combination with additive alpha-blending. The resulting density maps can either be visualized directly or used as an input to analysis functions, e.g., for outlier detection or real-time edge bundling [16]. Figure 9 depicts an exemplary density map created by the proposed visualization pipeline.

Post-processing Subsequent to the rasterization of the actual geometry, various post-processing stages are performed to apply additional effects to visualization artifacts and thus enhance the visual quality of the rendered images [32]. In this stage, image filters such as screen-space ambient occlusion [8] and unsharp masking the depth buffer [29] are applied to enhance depth perception in massive 3D trajectory visualization and to ease differentiation of single trajectories. Also, image-based anti-aliasing [27] is applied to alleviate aliasing

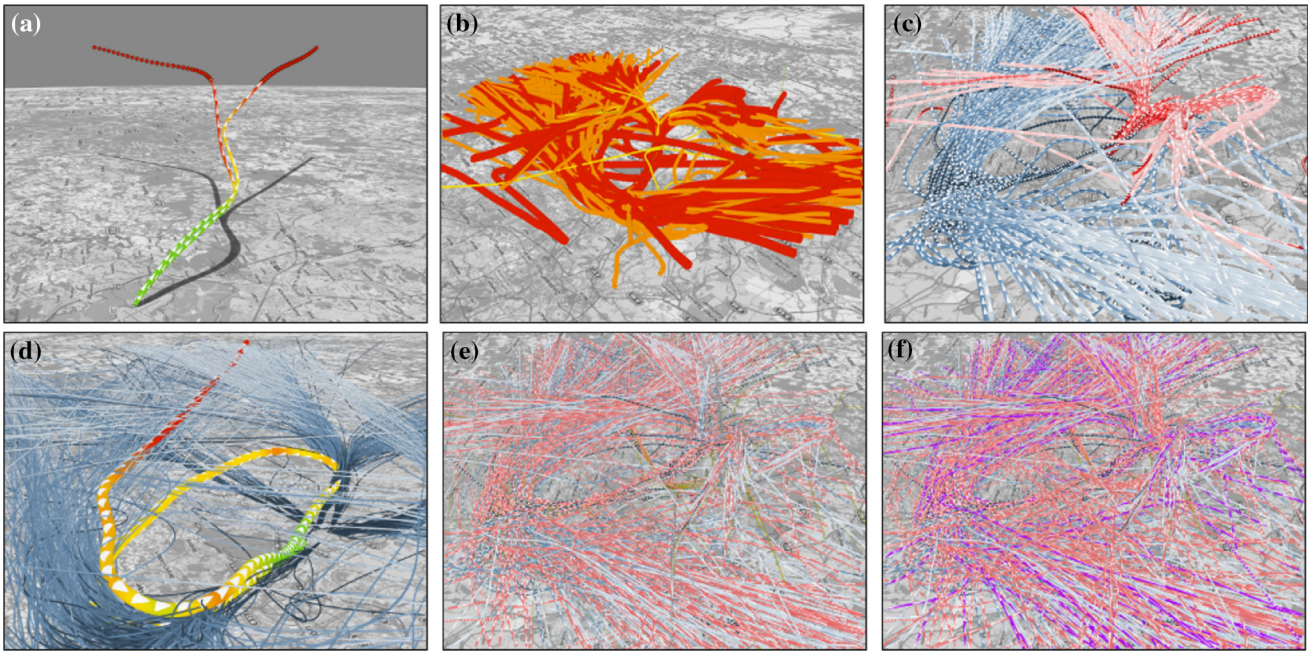


Fig. 7 Example applications using real-time animated 3D trajectory visualization. **a** Dynamic mapping of per-node attributes to visual properties. The trajectory is rendered as a tube in the foreground, in the rear, nodes are depicted by *spheres*. Speed is mapped to *color*. **b** Mapping of per-trajectory attributes to visual properties to visualize aircraft types. The weight classes are depicted by diameter and color (from *red* for large aircrafts to *green* for light aircrafts). **c** Visualization of approaches (*blue*) and departures (*red*). Texture mapping and animation are used to communicate direction (*arrow texture*) and speed (texture stretching and animation speed). **d** Highlighting of a trajectory representing a missed approach on an airport, rendered as a detailed tube with speed mapped to *color*. Unselected trajectories are displayed as thin untextured lines. **e** Temporal focus+context visualization to compare the trajectories of two different days. The two distinct temporal foci are depicted by *red* and *blue tubes*, the context is visualized by *yellow dots*. **f** Temporal focus+context visualization displaying two overlapping temporal foci (using *red* and *blue* for the two foci, violet for the overlapping area). The context is visualized by *yellow dots*

effects on the rendered edges. Another post-processing stage is responsible for applying highlighting to items that are currently selected by the user.

5 Application examples

This section gives application examples for the visualization of massive air-traffic data (Fig. 7) and describes the visualization and interaction techniques that are supported by the presented approach.

Attributed trajectory visualization As a fundamental component, the visualization of attributed 3D trajectories is supported. This includes visualization of per-node attributes (Fig. 7a), per-trajectory attributes (Fig. 7b), and the application of classifications to differentiate between multiple types of trajectories (Fig. 7c).

Object highlighting Single or multiple *highlighting* of specific trajectories (Fig. 7d) is a basic feedback functionality in interactive systems, e.g., to emphasize objects that are selected by a user, or to communicate computation results such as outlier detection (A.4) or cluster analysis (A.2). As a result of configuration or interaction, the identifiers of all trajectories to be highlighted are stored in a *highlighting buffer*, and a specific mapping configuration is chosen during mapping (Sect. 4.3), when a highlighted trajectory is rendered. Users can configure different highlighting modes, ranging from differentiating objects visually (e.g., by assigning different colors) to rendering objects in higher detail or displaying details only on highlighted objects (A.1).

Temporal Focus+Context visualization According to [9], the mapping approach can be applied to create temporal focus+context visualization. For that, all trajectories within a (user-defined) time interval are considered to be within the *temporal focus*, while the remaining trajectories represent the *context*. One or multiple temporal foci can be defined by manipulating a time interval in a dedicated user interface and selecting a distinct mapping configuration for the objects within that time interval. Another mapping configuration is used for the context (i.e., all unselected objects), and an additional configuration can be defined for objects in overlapping focus regions (i.e., objects that are associated with more than one focus interval).

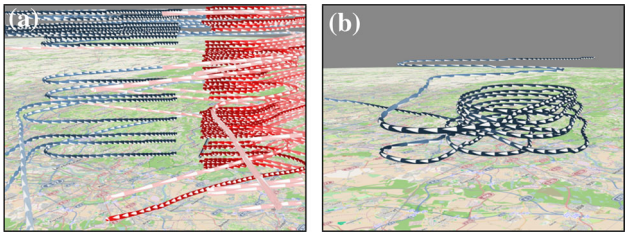


Fig. 8 Exemplary space–time cube visualization. **a** Temporal order of departing (*red*) and arriving (*blue*) aircrafts. **b** Detailed visualization of the movement of a single trajectory over time

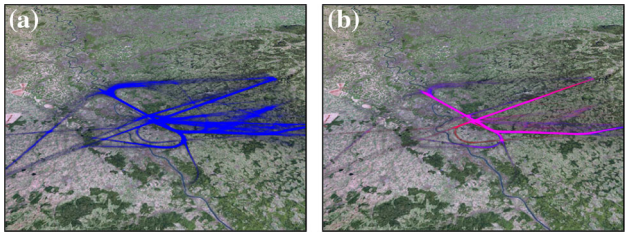


Fig. 9 Exemplary visualization of 2D density maps. **a** Aggregated view on aircraft movements over a single day. **b** Comparison of movement distributions between 2 days

Temporal focus+context visualization enables users to visually differentiate and compare the trajectories in different time intervals and the respective context (A.3), e.g., to estimate the variation in numbers of trajectories (C.1). In Fig. 7e, two such focus regions are defined at disjunctive time intervals. Trajectories within the focus areas are depicted in red and blue, respectively, while trajectories in the temporal context are depicted as yellow dots. Figure 7f shows a similar setting and additionally visualizes trajectories within the overlapping time interval using a third color (violet).

Space–time cube To express temporal relationships between trajectories at higher detail (C.1), e.g., to visually compare a number of trajectories (A.1, A.3), space–time cube visualization can be used. This can be achieved by applying a configuration which maps the time attribute to height. Figure 8a applies this to visualize the temporal order of departing and landing aircrafts, while Fig. 8b shows the movement of a single trajectory.

Density maps Density maps represent movement density, e.g., the traffic volume, at each spatial position. They can either be precomputed using a static configuration, or created in real time, to reflect current filter options and camera perspectives chosen by the user. Attribute mapping can be used to vary the influence of certain trajectories to the overall density, e.g., based on the speed or the weight class of an aircraft.

As a basic use case, density maps can be applied to assess the amount of traffic in a selected area and time interval (A.3) (Fig. 9a). Further, they can be used to easily compare traffic volumes in different situations (Fig. 9b) such as specific time intervals or different planning scenarios (A.5). Another application can be the detection of outliers (A.4), which can be achieved by comparing individual trajectories to the regions of high density in the resulting density map.

Detail+overview visualization Density maps can further be used for implementing detail+overview visualization [25]. Here, a density map constitutes the overview, and by selecting an area in the overview map, the contributing trajectories are highlighted in the detail view, which contains individual trajectory renderings (Fig. 10).

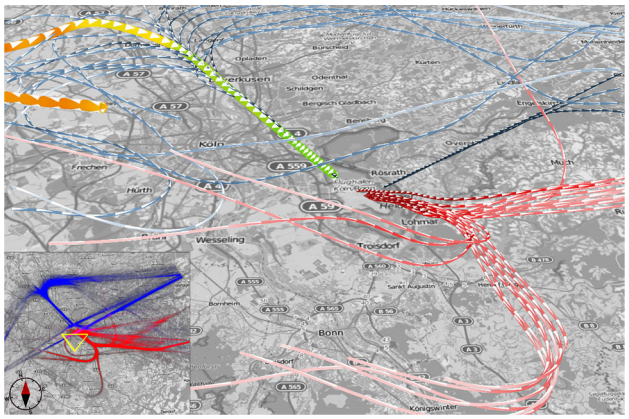


Fig. 10 Example detail+overview visualization, combining density maps with 3D trajectory rendering

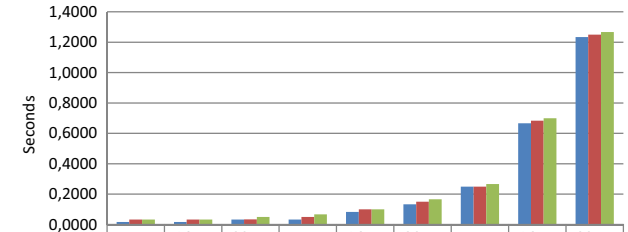
6 Discussion and performance evaluation

This section contains a performance evaluation of the approach and discusses limitations and shortcomings.

6.1 Performance evaluation

To evaluate the performance of our visualization technique, we used a data set of 10,000 aircraft trajectories. The performance test was conducted on a GeForce GTX 285 graphics board. Three variables were used for performance testing: (1) number of trajectories (100, 1,000, and 10,000), (2) tessellation level, i.e., the number of polygons rendered for each tube segment (4, 16, 32), and (3) screen size (800×600, 1280×960, and 1920×1080 pixels). Each of these variables were tested in combination, resulting in 27 individual tests.

The results shown in Fig. 11 indicate geometry generation as the main bottleneck of the implementation. Rendering time increases approximately linearly with the number of trajectories and tessellation level. For a small number of trajectories, higher tessellation levels have almost no impact



	4 TL	16 TL	32 TL	4 TL	16 TL	32 TL	4 TL	16 TL	32 TL
800x600	0,0166	0,0166	0,0333	0,0333	0,0833	0,1333	0,2497	0,6666	1,2333
1280x960	0,0333	0,0332	0,0335	0,0499	0,0999	0,1499	0,2499	0,6833	1,2499
1920x1080	0,0333	0,0332	0,0499	0,0666	0,0999	0,1666	0,2666	0,6999	1,2664

Fig. 11 Performance evaluation of the pipeline with respect to three different numbers of trajectories, screen resolutions, and tessellation levels (TL)

on rendering time. A higher viewport resolution has only a small performance impact, indicating that the technique is not fillrate-limited.

6.2 Potentials for future research

To increase the supported size of data, out-of-core and streaming algorithms should be applied, also techniques that simplify the geometry of 3D trajectories could be implemented into the mapping process. The concept of density maps can be extended to 3D density volumes, to visualize and analyze movement density in 3D space.

An application of stereoscopic rendering, e.g., for immersive environments such as CAVEs, can help enhance the perception of 3D trajectories. Automatic analysis algorithms, e.g., to identify missed approaches, can be implemented into the GPU-based pipeline. Finally, concepts such as the time wave [26] can be applied, to better reflect the linear and cyclic characteristics of temporal data and to improve temporal exploration.

7 Conclusions

In this paper, we have presented a real-time animated visualization technique for 3D movement trajectories. Its efficiency results from performing all stages—filtering, mapping, and rendering—on the GPU. As a consequence, it boosts key functionality for visual analytics such as spatio-temporal filtering, and appearance configuration.

We have demonstrated its applications by several visualization and interaction techniques such as 3D trajectory visualization, temporal focus+context, real-time computation of density maps, and space–time cube visualization. Using a data set of large, attributed 3D aircraft trajectories, we have outlined the application of the technique for visual analytics tools in the context of air-traffic management.

As a general-purpose technique, furthermore, it can be adapted to 3D movement visualization in other domains as well. This technique, therefore, can serve as a building block for visual analytics applications and systems that require to interactively visualize large, complex spatio-temporal 3D movement data.

Acknowledgments This work was funded by the German Federal Ministry of Education and Research (BMBF) in the InnoProfile Transfer research group “4DnDVis”. We also wish to thank Deutsche Flugsicherung GmbH for providing the used data set.

References

1. Akenine-Möller, T., Haines, E., Hoffman, N.: Real-Time Rendering, 3rd edn. A. K. Peters Ltd, Natick (2008)
2. Andrienko, G., Andrienko, N.: Interactive cluster analysis of diverse types of spatiotemporal data. ACM SIGKDD Explor. Newsl. **11**(2), 19–28 (2010)
3. Andrienko, G., Andrienko, N., Rinzivillo, S., Nanni, M., Pedreschi, D., Giannotti, F.: Interactive visual clustering of large collections of trajectories. In: IEEE Symposium on Visual Analytics Science and Technology pp. 3–10 (2009)
4. Andrienko, G., Andrienko, N., Schumann, H., Tominski, C.: Visualization of trajectory attributes in space-time cube and trajectory wall. In: Cartography from Pole to Pole, pp. 157–163. Springer, New York (2014)
5. Andrienko, G., et al.: Space, time and visual analytics. Int. J. Geogr. Inf. Sci. **24**(10), 1577–1600 (2010)
6. Andrienko, N., Andrienko, G.: Exploratory Analysis of Spatial and Temporal Data: A Systematic Approach. Springer, New York (2005)
7. Andrienko, N., Andrienko, G., Gatalsky, P.: Visual data exploration using space-time cube. In: 21st International Cartographic Conference, pp. 1981–1983 (2003)
8. Bavoil, L., Sainz, M.: Screen space ambient occlusion. NVIDIA **6** (2008)
9. Carvalho, A., de Sousa, A.A., Ribeiro, C., Costa, E.: A temporal focus + context visualization model for handling valid-time spatial information. Inf. Vis. **7**(3), 265–274 (2008)
10. Chang, R., Ebert, D., Keim, D.: Introduction to the special issue on interactive computational visual analytics. ACM Trans. Interact. Intell. Syst. (TiiS) **4**(1), 3 (2014)
11. Elmqvist, N., Tsigas, P.: A taxonomy of 3D occlusion management for visualization. IEEE TVCG **14**(5), 1095–1109 (2008)
12. Hägerstrand, T.: What about people in regional science? In: Papers of the Regional Science Association, pp. 7–21 (1970)
13. Hurter, C., Alligier, R., Gianazza, D., Puechmorel, S., Andrienko, G., Andrienko, N.: Wind parameters extraction from aircraft trajectories. Computers, Environment and Urban Systems (2014)
14. Hurter, C., Conversy, S., Gianazza, D., Telea, A.: Interactive image-based information visualization for aircraft trajectory analysis. Transp. Res. Part C Emerg. Technol. (2014)
15. Hurter, C., Ersoy, O., Fabrikant, S., Klein, T., Telea, A.: Bundled visualization of dynamic graph and trail data. IEEE TVCG (2013)
16. Hurter, C., Ersoy, O., Telea, A.: Graph bundling by kernel density estimation. In: Computer Graphics Forum, vol. 31, pp. 865–874. Wiley Online Library (2012)
17. Hurter, C., Tissoires, B., Conversy, S.: Fromdady: spreading aircraft trajectories across views to support iterative queries. IEEE TVCG **15**(6), 1017–1024 (2009)

18. Kessenich, J., Baldwin, D., Rost, R.: The OpenGL Shading Language Language Version: 4.40 Document Revision 9. The Khronos Group Inc. (2014)
19. Klein, T., van der Zwan, M., Telea, A.: Dynamic multiscale visualization of flight data. In: VISAPP 2014 (2014)
20. Kraak, M.J.: The space-time cube revisited from a geovisualization perspective. In: Proceedings of 21st International Cartographic Conference, pp. 10–16 (2003)
21. Kraak, M.J., Koussoulakou, A.: A visualization environment for the space-time-cube. In: Developments in Spatial Data Handling: 11th International Symposium on Spatial Data Handling, pp. 189–200. Springer, New York (2005)
22. Kristensson, P.O., et al.: An evaluation of space time cube representation of spatiotemporal patterns. IEEE TVCG **15**(4), 696–702 (2009)
23. Krone, M., Bidmon, K., Ertl, T.: Gpu-based visualisation of protein secondary structure. In: TPCG'08, pp. 115–122 (2008)
24. Kveladze, I., Kraak, M.J., van Elzakker, C.P.: A methodological framework for researching the usability of the space-time cube. Cartogr. J. **50**(3), 201–210 (2013)
25. Leung, Y.K., Apperley, M.D.: A review and taxonomy of distortion-oriented presentation techniques. ACM Trans. Comput. Hum. Interact. **1**(2), 126–160 (1994)
26. Li, X., Kraak, M.J.: The time wave. a new method of visual exploration of geo-data in time-space. Cartogr. J. **45**(3), 193–200 (2008)
27. Lottes, T.: Fxaa (2009)
28. Luebke, D.P.: Level of Detail for 3d Graphics. Morgan Kaufmann (2003)
29. Luft, T., Colditz, C., Deussen, O.: Image enhancement by unsharp masking the depth buffer. ACM Trans. Graph. **25**(3), 1206–1213 (2006)
30. MacEachren, A.M.: How maps work: representation, visualization and design. Guilford Press (1995)
31. Nienhaus, M., Döllner, J.: Depicting dynamics using principles of visual art and narrations. IEEE CGA **25**(3), 40–51 (2005)
32. Saito, T., Takahashi, T.: Comprehensible rendering of 3-d shapes. ACM SIGGRAPH **24**(4), 197–206 (1990)
33. Scheepens, R., Willems, N., van de Wetering, H., van Wijk, J.J.: Interactive Density Maps for Moving Objects. IEEE CGA **32**(1), 56–66 (2012)
34. Sidharth, T., Hanson, A.: A 3D visualization of multiple time series on maps. In: 14th International Conference Information Visualisation, pp. 336–343 (2010)
35. Tominski, C., Schulz, H.J.: The great wall of space-time. In: Workshop on Vision, Modeling & Visualization (VMV), pp. 199–206. Eurographics Association (2012)
36. Tominski, C., Schulze-Wollgast, P., Schumann, H.: 3D information visualization for time dependent data on maps. In: Ninth International Conference on Information Visualisation (IV'05), pp. 175–181
37. Tominski, C., Schumann, H., Andrienko, G., Andrienko, N.: Stacking-based visualization of trajectory attribute data. IEEE TVCG **18**(12), 2565–2574 (2012)
38. Trapp, M., Schmechel, S., Döllner, J.: Interactive rendering of complex 3d-treemaps with a comparative performance evaluations. GRAPP IVAPP **2013**, 165–175 (2013)
39. Ware, C.: Information visualization, vol. 2. Morgan Kaufmann (2000)
40. Willems, N., van de Wetering, H., van Wijk, J.: Visualization of vessel movements. Computer Graph. Forum **28**(3), 959–966 (2009)

Stefan Buschmann studied computer science at the University of Braunschweig, Germany (2000-2007). In 2011, he joined the Computer Graphics Systems group at the Hasso Plattner Institute, Potsdam, as a research assistant and PhD student. His research areas are 3D computer graphics, visual analytics, and spatio-temporal data visualization.

Matthias Trapp studied computer science at the Hasso Plattner Institute / University of Potsdam in Germany. He also received here his Ph. D. in computer science. Currently, he is leading the junior research group of the InnoProfile-Transfer-Initiative "4D-nD Geovisualization" at the Hasso-Plattner Institute / University of Potsdam in Germany. His major research areas are computer graphics, geovisualization, and software visualization.

Jürgen Döllner studied mathematics and computer science at the University of Siegen, Germany (1987-1992). He got his Ph.D. in computer science from the University of Münster, Germany, in 1996; he also received here his habilitation degree in 2001. In 2001 he became full professor for computer science at the Hasso-Plattner-Institute at the University of Potsdam, where he is leading the computer graphics and visualization department.